## CSE 322 Socket Programming Online – Mail Server

Suppose there is a Gmail server which stores **Inbox** of every client. An Inbox is an array of message. A message is a pair of (sender, body). A client can view his Inbox. Also, he can send message to anyone. Server assigns a unique ID to every. You have to implement these logics using Java Socket programming.

When server establishes a new connection with a client, it assigns an integer ID to the client and informs the client about his ID. Client prints the ID in his console ("I am client 1") to inform the user using the client program. First client receives ID=1, second client receives ID=2 and so on. Default status of each client is "No status". **Server must handle each client via a different thread**.

After establishing connection and receiving ID (myID), client will prompt from user command from console (`Scanner sc = new Scanner(System.in); sc.nextLine()`). A user has two command - inbox and send.

After **inbox** command, client will communicate with server. Server will first send the number of messages in client's inbox. Then server will send messages one-by-one from the inbox of the client. Client will receive all the messages and show them in console as following format.

```
<senderid>: <body>
```

The **send** command has two parameters, an integer **(receiver)** and a String **(body)**. After this command and parameters, client will send both of these parameters to server. Server will store the message as a pair **(sender, body)** in the **Inbox** of **receiver**. When receiver client gives **inbox** command to server, he will able to view the message (sender and the body).

**Implementation:**

Server side: Server.java, Worker.java, Message.java (if you want to store sender and body together as an object)

Client side: We have 2 clients - Client1.java, Client2.java

Or, you can run Client.java 2 times

**Warnings:**

- Match strings using ".equals()" method
- Don't use Hashmap, it is not thread safe. Use Hashtable instead.
- Open OutputStream first, always
- Open buffers only one time for each socket